EVOLUTIONARY PATH PLANNING FOR AUTONOMOUS AIR VEHICLES USING MULTIRESOLUTION PATH REPRESENTATION

Ravi Vaidyanathan[1,3*], Cem Hocaoglu[2], Troy S. Prince[1], Roger D. Quinn[3]
[1]Orbital Research Inc., Cleveland, OH, USA
[2] State University of New York at Stony Brook, Stony Brook, NY, USA
[3]Case Western Reserve University, Cleveland OH, USA

Abstract

There is a recognized need for automated path planning for unmanned air vehicles (UAVs) and guided munitions. Evolutionary programming approaches provide an alternative to classical functional optimization methods with the capability of incorporating a variety of optimization goals, while tolerating vehicle constraints. In this work, we introduce an evolutionary flight path planning algorithm capable of mapping paths for free-flying vehicles functioning under several aerodynamic constraints. An air-to-ground targeting scenario was selected to demonstrate the algorithm. The task of the path planner was to generate inputs flying a munition to a point where it could fire a projectile to eliminate a ground target. Vehicle flight constraints, path destination, and final orientation were optimized through fitness evaluation and iterative improvement of generations of candidate flight paths. Evolutionary operators comprised of one crossover operation and six mutation operators. Several cases for air-to-ground vehicle targeting have been successfully executed by the evolutionary flight path planning algorithm under challenging initial conditions. A feasible path is typically found rapidly (<100 generations), with further optimization (~3000 generations) insuring a strike very near target center. These results clearly demonstrate that evolutionary optimization using can achieve flight objectives for air vehicles without violating limits of the aircraft.

I. Introduction

Autonomous free-flying entities such as unmanned aircraft, cruise missiles and other precision guided air vehicles are increasingly dependent on automatic control. One highly relevant scenario is the need for automated path planning within critical vehicle constraints for unmanned air vehicles (UAVs) and guided munitions for target-seeking and/or goal orientation operations. Necessary features of such a system include: rapid autorouting capability, incorporation of vehicular and environmental constraints into path generation, and the ability to efficiently perform path planning functions within strict time constraints [1].

Current path planning technologies do not fully achieve future demands for guided air vehicles [1,2]. Specific deficiencies include:

- Many autorouters only address higher-level path planning behavior; vehicular dynamics and constraints are not fully considered [1,2]
- Many existing path planning methods are of little use in situations involving vehicle constraints, resulting in bottle-necks with many local minima [3],
- The need for autorouting for 6 degree of freedom (dof) platforms under strict vehicle constraints has not been fully addressed [4],
- Conventional path planners typically use a fixed path representation; such methods cannot self-adjust to problem complexity [5],
- Accounting for time-varying or non-holonomic constraints in dynamic environments often hampers planner performance [5],
- Optimization criterion often are complex and subjective; many traditional planners optimize only with respect to the shortest path [5] without full consideration of critical constraints.

Global planning approaches [6,7] are in general complete in that if a path exists, it will be found, yet the computational expense grows with vehicle complexity. Although local methods such as potential field procedures [8,9] are more efficient, they can become trapped in local minima. In general, functional optimization methods have tendencies to become trapped within local minima, and may be difficult to implement on free-flying platforms since complexity grows exponentially with vehicle degrees of freedom [10]. Furthermore, enumerative techniques are susceptible to inefficiencies when dealing with complex vehicles due to size of search space.

A class of heuristic searching methods based upon simulated evolution known broadly as *Genetic Algorithms* (GA) has become very popular lately for discrete optimization problems characterized by many local minima in nondifferentiable, discontinuous or constrained problem spaces. These evolutionary techniques are population-oriented: successive populations of feasible solutions are generated in a stochastic manner following laws similar to that of natural selection. This contrasts standard programming techniques that normally follow a single trajectory repeatedly until a satisfactory solution is reached.

Many previous path planners cannot accommodate a variety of optimization criteria or allow changes in these standards without changing the characteristics of the planner or the search map. Evolutionary approaches, on the other hand, can handle a variety of optimization goals and are very tolerant to the form of the evaluation function. Functions to be optimized need not be

*corresponding author: raviv@orbitalresearch.com

differentiable or continuos. Evolutionary path planning approaches are also flexible to changes in environment and are robust to uncertainties.

Evolutionary approaches provide an alternative form of path generation capable of incorporating a variety of optimization goals, while tolerating vehicle constraints. These benefits have lead to the development of several evolutionary path planners (a brief summary is presented in[5]). In this work, our past research [5,11,12] in multi-dimensional path planning is expanded to create an *evolutionary flight-path planning algorithm* capable of mapping paths for free-flying vehicles functioning under aerodynamic constraints. The generation of flight trajectories for air-to-ground targeting for autonomous munitions is selected as a benchmark situation to demonstrate the utility of the path planning genetic algorithm.

The organization of this paper is as follows: in **Section II** the dynamics of a candidate autonomous air vehicle are modeled, **Section III** gives a description of the representation used for encoding paths, **Section IV** enumerates work performed within evolutionary evaluation function to incorporate aerodynamic constraints and flight objectives, **Section V** delineates the genetic operators of the path planning algorithm, **Section VI** summarizes the operation of the path planner for air-to-ground target seeking, **Section VII** presents results of the algorithm when implemented on a candidate aircraft, and **Section VIII** briefly discusses conclusions, ongoing, and future work.

II. Air Vehicle Model Development

The general equations of motion of a 6 degree of freedom (dof) rigid airframe may be described through Newton's Laws in terms of the nomenclature enumerated in **Table 1**:

$$\tilde{F} = m\left(\frac{\partial \dot{\tilde{s}}}{\partial t} + \tilde{\omega} \times \dot{\tilde{s}}\right) \qquad (1\text{ a, b})$$

$$\tilde{M} = \frac{\partial(\mathrm{I} \cdot \tilde{\omega})}{\partial t} + \tilde{\omega} \cdot \times (I \cdot \tilde{\omega})$$

Aerodynamic forces acting on an air vehicle, are often expressed in the form[13]:

$$\tilde{F}_A = [C_f(\tilde{z})][Q_f] \qquad (2\text{ a, b})$$

$$\tilde{M}_A = [C_m(\tilde{z})][Q_m]$$

Where both [C] matrices are dimensionless coefficients which are functions primarily of aircraft state $\tilde{z}$ = (V, α, β, *p*, *q*, *r*), and each [Q] is a product of flight dynamic pressure, and aircraft reference area or characteristic length, respectively. The system inputs, u(t), include aerodynamic forces developed by actuator deflections and propulsive forces, and environmental effects, whose impact on the air vehicle may be reflected in state space form as:

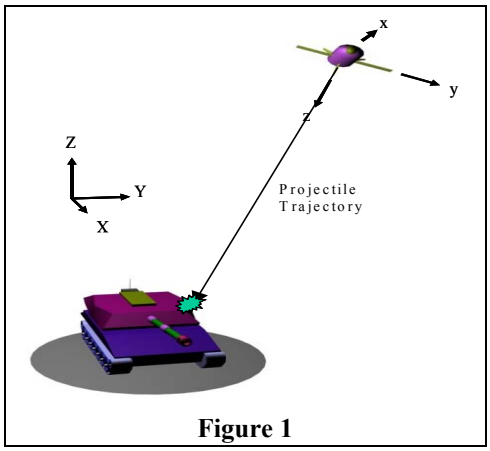

Figure 1

$$\dot{\tilde{z}} = A\tilde{z} + B\tilde{u} \qquad (3)$$

The essential mode of operation for air vehicle autopilot systems is to move control surfaces (δp, δq, δr) in response to desired roll rate (*p*), pitch rate (*q*), and yaw rate (*r*) commands based on a linearized airframe model [13].

The evolutionary algorithm also plans functions for the aircraft based upon a linear airframe response. The actual control inputs to the system (i.e. control surface deflections) are generated and modified to achieve desired targeting objectives. Additionally, since the algorithm also changes the time of each control input, the minimum time step is constrained by the frequency response characteristics of the actuators themselves. These inputs are integrated over time to produce the time history of the state variables, including aircraft roll, pitch and yaw rates. During operation, these rates form inputs to the autopilot, which directs the aircraft to follow the developed trajectory

Variable	Parameter Description
$\tilde{F}$	Total force vector acting on airframe
$\tilde{M}$	Total moment vector acting on airframe
$\tilde{s}$	Position vector of mass center of airframe
$\tilde{\omega}$	Angular velocity of airframe (body-fixed)
m	Air vehicle mass
I	Inertia matrix of air vehicle
δp, δq, δr	Elevator, aileron, and rudder deflections
V	Absolute vehicle airspeed (global)
u	Forward velocity (body centered)
v	Side velocity (body centered)
w	Downward velocity (body centered)
α	Angle of attack=$\tan^{-1}(w/u)$
β	Sideslip angle=$\tan^{-1}(v/u)$
p	Angular roll rate
q	Angular pitch rate
r	Angular yaw rate

Table 1 – Nomenclature

As a case study for evolutionary flight path generation, an air-to-ground targeting scenario was chosen. In this scenario, a small airborne autonomous munition fires a

kinetic energy projectile straight downward from its center of gravity to strike a ground target. The firing action of the munition is depicted in **Figure 1**, with global and body fixed reference frames delineated, along with the projectile trajectory. The task of the path planner was to generate a set of inputs which would fly the munition to a point where the projectile trajectory would strike as near as possible to the target center. Note that the angular orientation of the vehicle as well as its position is critical for proper target strike. Thus, unlike the majority of munition targeting where only global position at strike is relevant, this problem is a six dof optimization. Also, for realistic depiction of targeting scenarios, the aircraft was forced into "glide mode" (forward thrust could not be generated), to provide a state analogous to endgame situations where engines delay is unreliable to reach a target at close range.

To develop realistic flight model path planning tests, the Flight Dynamics and Control Analysis (FDC) 1.2 toolbox for the MatLab/Simulink environment was used [14] The FDC toolbox includes a complete non-linear model of a DHC-2 Beaver; a light, single engine, high wing aircraft. This model was modified to improve responsiveness, and more closely resemble the flight characteristics of autonomous airborne munitions. The linearized model comprised of equations (1), (2), and (3) was implemented within the genetic algorithm to evaluate system flight performance during evolutionary optimization. The planner modified control surface inputs, and time step between inputs (limited by actuator responsiveness) to produce an effective flight path to the target. In flight, roll, pitch, and yaw rates resulting from the control inputs may be fed directly to an autopilot to guide the air vehicle along the path.

III. Multiresolution Flight Trajectory Representation

A fundamental aspect of any heuristic optimization is the specific structure of encoded information enabling iterative progression towards an ideal solution. For air vehicle flight trajectory generation, a *multiresolution binary tree* representation is implemented such that complex paths of higher dimensionality to allow efficient evolutionary operation over generations of flight trajectories.

Multiresolution methods have been used in signal and image processing to provide efficient data representation adapted to the complexity of the signal content, as well as in our own past work [5,11,12] for path planning of autonomous robots with less stringent motion constraints. Furthermore, tree-like representations have seen extensive use in genetic programming [15], particularly in situations where mutation is relied on to play a significant role in evolution [16,17,18]. The algorithm implemented for automatic routing of autonomous air vehicles in this work utilizes a related approach to path representation. In addition to accurate path encryption, the representation can reduce expected search lengths for trajectory generation. If a successful path is found early in the search hierarchy further expansion of that portion is not necessary; the structure of the binary tree is thus internally optimized based upon problem complexity. This advantage is mapped into the search space and the string length is adjusted accordingly, enhancing precision and computational efficiency.

Within the binary tree, air vehicle flight paths are represented by hierarchically ordered nodes, each containing a specific set of control surface deflections, $\Delta i_n = (\delta p_n, \delta q_n, \delta r_n)$, accompanied by a time shift, Δt_n, to form a complete array of parameters, Δu_n, at each node. Δt_n represents the delay from the last input Δi_{n-1}, to the current input Δi_n. The tracking of all such nodes maps the total set of control inputs and time intervals piloting the air vehicle over the flight trajectory. These nodes are organized in a tree-like structure, as shown in the left section of **Figure 2**. The complete path from start to finish is reflected by this binary tree, with the n^{th} node corresponding to the input Δu_n, which is comprised of the control surface deflection Δi_n, occurring at the time $t = \sum_{j=0}^{j=n} \Delta t_j$.

The *in-order* traversal of this binary tree provides a sequence of knot points representing all control inputs over the flight trajectory from start to finish, as described by each P_i in the right section of **Figure 2**, shown with start (S) and end (G) points. Intermediate nodes in the figure are generated randomly, and may subsequently be perturbed, inserted, or removed to modify and optimize the path. The sequence of nodes defined by the bottom of **Figure 2** is:(S = start, G = goal):S→ P_4→ P_3→ P_5→ P_2→ P_6→ P_1→ P_8→ P_9→ P_7→ G.) Although the number of nodes varied from path to path, no particular constraints or limits based on size were implemented with regard to node selection for crossover or mutation.

The trajectories represented through multiresolution binary trees do not correspond directly to actual aircraft flight paths; rather *they decode into a set of piloting commands, which may be passed directly to an autopilot for the air vehicle to maneuver itself along the flight trajectory*. Large arrays of candidate flight trajectories

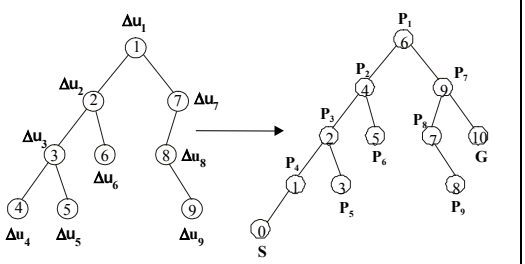

Figure 2

may be created rapidly using multiresolution path representation by generating and ordering random sets of control surface inputs (Δi), along with random time intervals (Δt) between inputs. Provided each Δi falls within acceptable control surface deflections, and each Δt is greater than or equal to the system actuator delay, the flight path will be achievable within vehicle actuator constraints. Vehicle flight dynamics and constraints, as well as path destination and final orientation, may be optimized through proper fitness evaluation and iterative improvement of collections of multiresolution flight paths.

IV. Evolutionary Fitness Evaluation

Given a generation of candidate multiresolution flight trajectories for air vehicle targeting, some method of measuring positive and negative aspects of paths is critical for the selection of paths for genetic union (crossover) and mutation. This measure must reflect both the validity of the flight path in relation to violations of the vehicle's flight envelope and maintenance of flight stability, and for target strike accuracy. This problem cannot be solved by a single performance measure.

Such situations are also commonly encountered outside the field of aircraft control; many real-world problems are composed of a set of variables that must be specified, and a set of constraints which restrict the those variables [19] . The concept of multiobjective evolutionary algorithms have been described [20,21,22,23] as a modification of the standard genetic algorithm at the selection level to attack such constraints, in particular for aircraft [24][i]. The main difference between a conventional evolutionary algorithm and a multiobjective genetic algorithm resides in fitness assignment [22]. In a review of evolutionary approaches to multiobjective optimization, Fonseca and Fleming [25], categorize multicriteria evolutionary algorithms and compare fitness assignment strategies. In particular, they distinguish plain aggregating approaches, population-based non-Pareto based approaches, and Pareto-based approaches.

Aggregation methods combine the objectives into a higher scalar function that is used for fitness calculation. Population aggregation methods include: the weighted-sum approach, target vector optimization, and the method of goal attainment [25]. Population-based non-Pareto approaches are able to evolve multiple nondominated solutions concurrently in a single simulation run. By changing the selection criterion during the reproduction phase, the search is guided in several directions simultaneously [25]. Pareto-based fitness was proposed in [26]. All approaches of this type use Pareto dominance in order to determine the reproduction probability of each individual [23]. A major advantage of the Pareto-based fitness approach is the lack of sensitivity to nonconvexity in Pareto-optimal sets [25]. An aggregation method of assigning fitness scoress was implemented in this work, primarily due to the fact that only a single solution was desired. Three fitness stages were summed for scalarization.

Prior to fitness assignment actuator commands represented by the binary tree must first be decoded into a mapping of the flight motion of the aircraft. The decoding process begins from the air vehicle initial state (z_0), to the time (Δt_1) of the first actuator input (Δi_1) by: 1) solving and integrating equation (3) from $t=0\rightarrow\Delta t_1$ with input $u=\Delta i_0$ to obtain all aircraft states (z) over the interval, 2) solving equations (2 a & b) using the derived states (z) to determine all applied forces (F) and moments (M) from $t=0\rightarrow\Delta t_1$, and 3) solving and integrating equation (1 a & b) to obtain a global map of all aircraft translational positions (s) and angular orientations (ω) from $t=0\rightarrow\Delta t_1$. The end result of this decoding is the set of all states, global positions, and angular orientations of the aircraft following the flight trajectory from starting position to the first node on the multiresolution path. This process is repeated from node 1 to node 2 for $t=\Delta t_1\rightarrow\Delta t_1+\Delta t_2$, with the actuator input $u=\Delta i_1$, and continues until the interval $t=\sum_{j=1}^{j=n-1}\Delta t_j \rightarrow \sum_{j=1}^{j=n}\Delta t_j$ with input $u=\Delta i_{n-1}$ is reached covering node (n-1) to node n. In the final decoding step, the actuator delay, Δt_{min} is added to the time at node n to provide integration limits from node n to end node G for the final input Δi_n, hence providing a map of aircraft transnational positions, angular orientations, and flight states over the entire multiresolution flight path.

Based upon the generated flight map, the aircraft path planner performs evolutionary fitness evaluation in three stages for the inclusion of all vital optimization criteria: **Stage 1** evaluates flight paths purely in relation to maintaining flight stability and avoiding flight envelope violations, **Stage 2** assesses path fitness in terms of non-critical flight parameters; *i.e.* constraints whose violation will not cause loss of aircraft control, yet whose limits ideally should not be exceeded, while **Stage 3** analyzes the path with respect to flight objectives by combining three factors: 1) proximity of path to desired goal, 2) orientation of aircraft, and 3) time of flight. The total evolutionary fitness for a given path is the weighted sum of scores from each stage, each of which is scaled according to its importance to flight goals.

In its implementation for ground vehicle targeting, a flight envelope consisting of limits on all flight parameters (V, α, β, p, q, r, ψ, θ, ϕ, x_e, y_e, H) was developed based upon the aircraft's dynamics; flying within these limits maintains flight control and stability. **Stage 1** of fitness evaluation analyzed the decoded flight map for violations of these limits providing fitness score S1:

[i] Note that this work focused on wing design as opposed to path planning

$$S_1 = \left[\sum_{i=1}^{i=k} V_i \right] * \rho_1 \qquad (4)$$

where the summation represents the total number of states whose limits were exceeded, k is the amount of states, and ρ_1 is a scaling constant for normalization. For these experiments, the only relevant parameter in **Stage 2** fitness were altitude limits given by:

$$S_2 = [V] * \rho_2 \qquad (5)$$

where the quantity in brackets reflects flight path altitude violations, and ρ_2 provides normalization. The air vehicle ideally should not infringe these altitude limits (e.g. too low exposes the plane to ground fire, too high causes uncertainty in targeting), but in the event that no alternative options exist, these limits may be exceeded.

Three factors comprised the **Stage 3** fitness:

$$S_3 = D(p) * \rho_{31} + O(p) * \rho_{32} + T(p) * \rho_{33} \qquad (6)$$

D(p) measures the distance (Euclidean norm) from the targeting point of the vehicle to the center of the target.

$$D(p) = \left\| \tilde{f} - \tilde{c} \right\| \qquad (7)$$

where c is a vector containing the target center coordinates, and f is a vector of the current target of the aircraft given by:

$$\tilde{f} = \begin{bmatrix} x_{e_f} + H_f * (\tan\hat{\theta}_f \cos\hat{\psi}_f + \tan\hat{\phi}_f \sin\hat{\psi}_f) \\ y_{e_f} + H_f * (\tan\hat{\phi}_f \cos\hat{\psi}_f + \tan\hat{\theta}_f \sin\hat{\psi}_f) \end{bmatrix} \qquad (8)$$

The subscript f indicates states at the final point of flight and the ^ symbol represents the Euler angles transformed into angular orientations on the body fixed frame.

Since a hit from directly above the target is preferable to one at an angle, O(p) is included in **Stage 3** as:

$$O(p) = \left[\tan^{-1} \left(\frac{\left\| \tilde{d} \right\|}{H} \right) \right]^{-1} \qquad (9)$$

where d is the vector difference between the target center and the point directly under the aircraft projected onto the plane of the target. O(p) thus approaches 0 as the angle or the target trajectory approaches 90°. The final fitness measure in **Stage 3** T(p) is simply the normalized time of flight. Note that ρ_{31}, ρ_{32}, and ρ_{33} are scaling constants.

Therefore, for a path (p) represented by an ordered set of nodes from beginning to end, the total evolutionary fitness score, F(p) is defined as scalar the sum of each of the stages, or:

$$F(p) = S_1 + S_2 + S_3 \qquad (10)$$

V. Evolutionary Operators

The use of multiresolution representation with variable length encoding for air vehicle trajectories introduces many unique opportunities for evolutionary operation with respect to the binary tree structures. Given that each node Δu, represents a set of control inputs occurring at a specified time with respect only to the preceding node, intermediate nodes may be freely interchanged with those from binary trees of other candidate trajectories or perturbed within the same tree without any loss of generality in path representation. Evolutionary operators utilized by the flight path planner expanded upon our past work[5] to generate one crossover operator, five random mutation operators, and one "intelligent" mutation operator suitable for aircraft targeting trajectory optimization. Specific operations consisted of:

- **Swap-Subtree Crossover**: combines two paths to reproduce two new ones by selecting a node at random from each parent, and exchanging subtrees branching from that node in their offspring. **Figure 3** delineates crossover operation, (one parent in white; its mate in black), where the "X" marked nodes are selected..

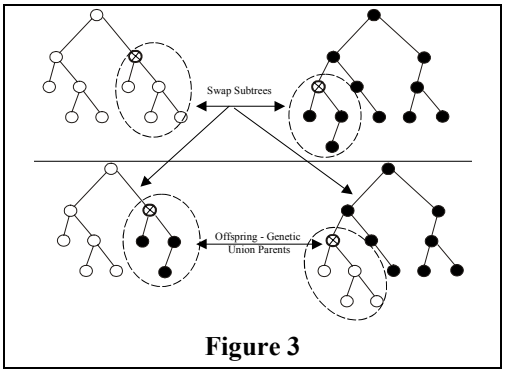

Figure 3

- **Perturb 1 Mutator**: randomly selects a node (n) in the binary tree and perturbs its contents (Δu_n) a small amount. This operator allows for fine tuning of acceptable, but not ideal, paths.
- **Perturb 2 Mutator**: randomly selects a node (n) in the binary tree and perturbs its contents (Δu_n) a large amount. This mutation makes significant alterations in flight trajectory, ideally to change an infeasible path into a feasible one, or to move an inaccurate path much closer to its target.
- **Swap-Node Mutator:** exchanges the contents (Δu) of two randomly picked nodes in the binary tree, shown in **Figure 4**.
- **Insert-Node Mutator**: creates a new intermediate path input (Δu), by inserting a node into the binary tree, shown in **Figure 5**.
- **Flip Mutator**: changes the sign of Δi within a randomly picked node
- **Fix Mutator** ("intelligent" mutation): operates in a similar manner to the Perturb 2 Mutator, except that its

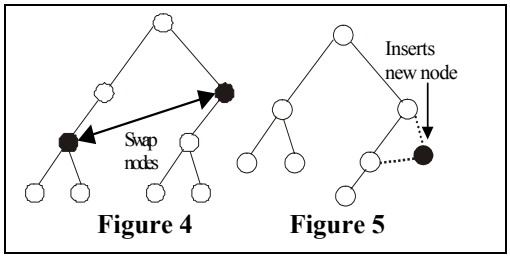

Figure 4 **Figure 5**

operation is not random. Within the selected binary tree, inputs Δu are changed specifically in relation to states violating flight constraints. For example, a binary tree mapping a flight path with too great a pitch angle, will be fixed by manipulating deflections within Δi such that θ is reduced for that section of the flight. The operation of this mutation is designed to fix input/output relationships of completely infeasible paths by altering the input and time of input, forcing the path into feasible regions.

VI. Evolutionary Target Seeking Implementation Process

Figure 6 outlines the implementation process for the evolutionary generation of ground targeting trajectories for air vehicles. Since certain evolutionary operators (**Section V**) are best suited to correcting infeasible paths, while others optimize feasible paths for greater accuracy, selection probabilities of each are adjusted once a target strike is achieved. The probability for each mutation was selected by trial and error. Two sets of probabilities were selected, one reflecting the state of evolution before a path striking the target was achieved, and one reflecting the state after a single successful path was achieved. A simple binary shift to the second set of probabilities was performed when a single successful path was achieved.

VII. Implementation Results

Utilizing the fixed wing aircraft model, evolutionary fitness scoring routines, and operators, the multiresolution aircraft path planning algorithm was implemented for optimal trajectory generation for ground target seeking. The movements of three control surfaces (elevator, aileron, and rudder) were optimized to develop a trajectory for the vehicle to reach a position such that its firing trajectory would strike a circular land based target of 2.5m radius.. Elementary targeting scenarios were run for very simple targeting tasks in a LINUX C++ programming environment. Following these simple cases, the air vehicle was assigned to strike targets that the air vehicle could not reach without optimizing its angular orientation along with its global position (i.e. the target was located in a position that the air vehicle could not fly directly above). Delay was set at Δt_{min}=0.15 sec, with input allowable at t=0. Sample results for initial populations varying from 50 to 70 individuals, a mutation probability of 0.65, and a crossover probability of 0.35 are shown in **Figures 7 to 10**.

Figure 7 shows frontal, lateral, and aerial views of critical munition positions along the path evolved to strike a target 30m directly in front (X dimension) and 100 m below (Z dimension) the air vehicle's initial position. Dashed lines in the lateral and top views show the projectile trajectory. For this situation, the path planner created a trajectory piloting the vehicle to an orientation allowing accurate target strike (0.027m from target center) in 0.59 sec of flight time. The angular orientations of the aircraft through flight (in radians) versus time are plotted in the lower right corner, while a logarithmic (base 10) plot of the best candidate in each generation versus the generation count is shown in the lower left (the flat line indicates the minimum fitness score necessary to strike the target).

Figures 8, **9**, and **10** show successful targeting paths for varying goal positions lateral to the aircraft (Y dimension). The target is not shown in the aerial views in **Figures 9** and **10** for better visualization, although the viewable portion of the projectile trajectory is still shown. Specific information for each run is enumerated at the top of each figure. In each of these runs, the target position forced the angular orientations allowing accurate target strike; most notably in **Figure 10** where roll and pitch angles are combined to achieve a firing trajectory 0.01m off a target center 10m lateral to plane starting position.

VIII. Conclusions

The results presented in **Section VII** demonstrate that evolutionary flight path optimization using multiresolution representation can achieve flight objectives without violating limits of the aircraft. Several cases for air-to-ground vehicle targeting have been successfully executed under challenging initial conditions. In most situations, a feasible path is found rapidly (<100 generations), although further optimization is necessary to insure a strike very near target center. To our knowledge, this is the only work of its kind optimizing all rotational and translational degrees of freedom of an air-to-ground munition for target strike. Future work plans include

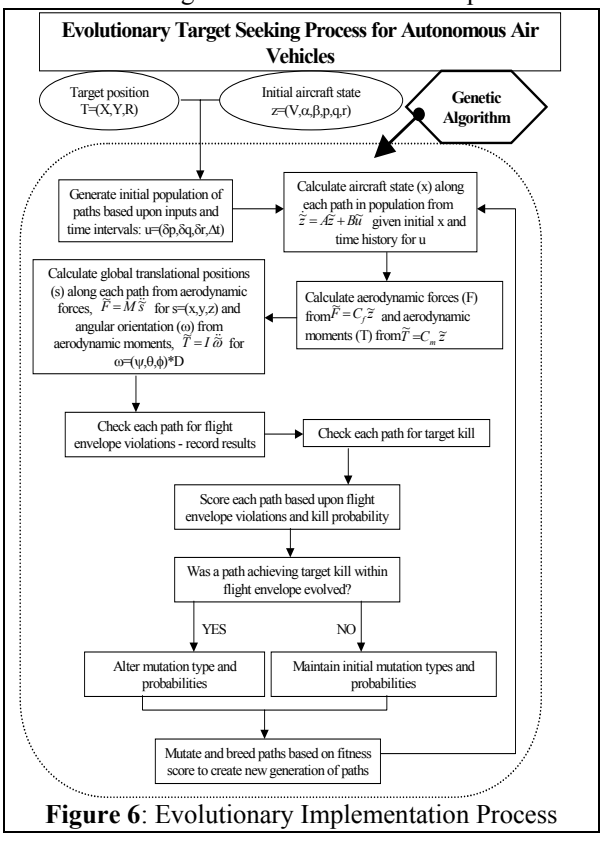

Figure 6: Evolutionary Implementation Process

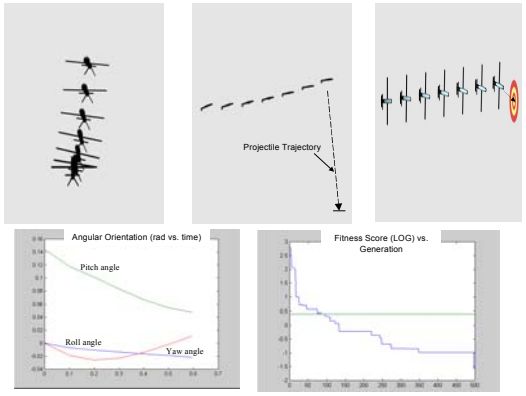

Figure 7

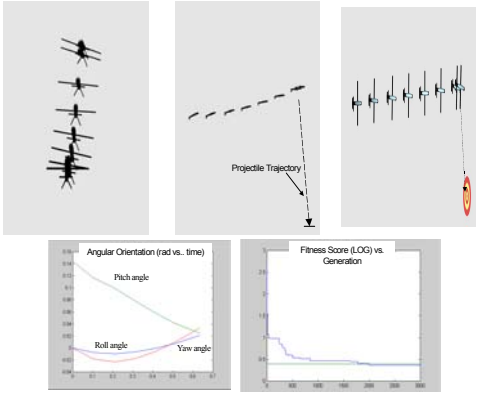

Figure 8

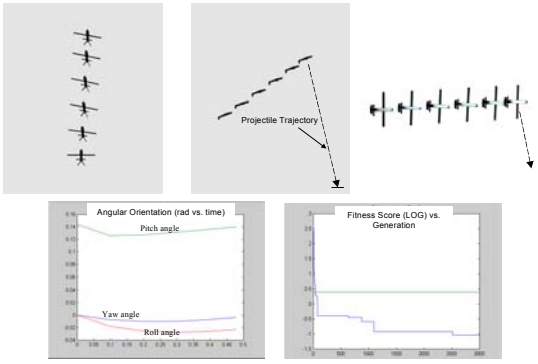

Figure 9

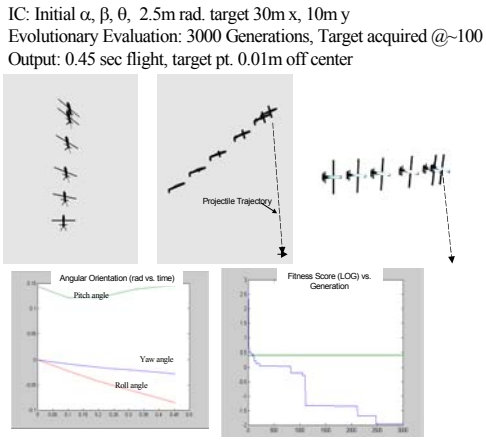

Figure 10

planning for target motion through optimization of intercept point, while current efforts involve the training of neural networks to reproduce generated flight trajectories for instantaneous reaction.

Acknowledgements

The authors gratefully acknowledge the support of Mr. Johnny Evers of The Air Force Research Laboratory Munitions Directorate, Orbital Research Inc., and Dr. Arthur Sanderson for facilitating contact between our research groups.

References

[1] Harman, W.L., "Strike Mission: Real-Time Retargeting (RTR) Planning and Optimization", in *Precision Strike Mission Area*, presentation at the Naval Air and Surface Weaponry Technology BAA Industry Day, Loftus, T., 1998

[2] Helgason, R.V., Hayasuriya, A.C., Kennington, J.L., Lewis, K.H., "Shortest Path Algorithms on Grid Graphs with Applications to Strike Planning", Technical Report to the Office of Naval Research, 1997

[3] Koren, Y., Borenstein, J., "Potential Field Methods and their Inherent Limitations for Mobile Robot Navigation", in *Proc. IEEE Int. Conf. on Robotics & Automation*, 1991

[4] Henrich, H.,Wurll, C., Worn, H., "6 DOF Path Planning in Dynamic Environments – A Parallel on-line Approach", in *Proc. IEEE Int. Conf. on Robotics & Automation*, 1998

[5] Hocaoglu, C., Sanderson, A.C., "Planning Multiple Paths with Evolutionary Speciation", *IEEE Tran. on Evolutionary Computation*, due in press, 2001

[6] Sharir, M., "Algorithmic Motion Planning in Robotics", in *IEEE Symp. on Robotics & Automation*, 1989

[7] Lozzano-Perez, T., "A Simple Motion Planning Algorithm for General Robot Manipulators", *IEEE Journal of Robotics & Automation*, 3, 1987

[8] Khosla, P., Volpe, R., "Superquadratic Artificial Potentials for Obstacle Avoidance and Approach", in *Proc. IEEE Int. Conf. on Robotics & Automation*, 1988

[9] Rimon, E., Doditschek, D.E., "Exact Robot Navigation using Artificial Potential Fields", *IEEE Trans. on Robotics & Automation*, 8, 1992

[10] Latombe, J., *Robot Motion Planning*, Kluwer Academic Publishers, 1991

[11] Hocaoglu, C., Sanderson, A.C., "Evolutionary Path Planning using Multiresolution Path Representation", *Proc. of IEEE Int. Conf. on Robotics & Automation*, 1998

[12] Hocaoglu, C., Sanderson, A.C., "Multimodal Function Optimization using Minimal Representation

Size Clustering and its Application to Planning Multi-Paths", *Evolutionary Computation Journal*, 5, 1997
[13] Nelson, Robert C., *Flight Stability and Automatic Control*, Mc Graw Hill, 1989
[14] Rauw, M., *FDC 1.2 –A SIMULINK Toolbox for Flight Dynamics and Control Analysis*, 1998
[15] Iba, H., "Random Tree Generation for Genetic Programming", in *Late Breaking Papers at the Genetic Programming 1996 Conference*, Koza, J.R. Ed., Stanford Univ. Bookstore, 1996
[16] Angeline, P.J., "Subtree Crossover: Building Block Engine or Macromutation?", in *Proc. Second Annual Genetic Programming Conf.*, 1997
[17] Luke, S., Spector, L., "A Comparison Crossover and Mutation in Genetic Programming", in *Proc. Second Annual Genetic Programming Conf.*, 1997
[18] Chellapilla, K., "Evolving Computer Programs Without Subtree Crossover", *IEEE Trans. On Evolutionary Computation*, 3, 1997
[19] Dozier, G., Boen, J., Homaifar, H., "Solving Constraint Satisfaction Problems using Hybrid Evolutionary Search", *IEEE Transactions on Evolutionary Computation*, 2, 1998
[20] Fonseca, C.M., Fleming, P.J., "Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion, and Generalization", *Genetic Algorithms: Proceedings of the Fifth International Conference*, 1993
[21] Srinivas, N., Deb, K., "Mulitobjective Optimization using Nondominated Sorting in Genetic Algorithms", *Evolutionary Computation*, 2, 1994
[22] Fonseca, C.M., Fleming, P.J., "Multiobjective Genetic Algorithms", available through University of Sheffield Department of Automatic Control and Systems Engineering, Sheffield, S1 4DU, United Kingdom, 1994
[23] Zitzler, E., Thiele, L., "Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach", *IEEE Transactions on Evolutionary Computation*, 3, 1999
[24] Obayashi, S., Sasaki, D., Takeguchi, Y., Hirose, N., "Multiobjective Evolutionary Computation for Supersonic Wing-Shape Optimization", *IEEE Transactions on Evolutionary Computation*, 4, 2000
[25] Fonseca, C.M., Fleming, P.J., "An Overview of Evolutionary Algorithms in Multiobjective Optimization", *Evolutionary Computation*, 3, 1995
Goldberg, D.E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addision-Wesley, 1989
[26] Goldberg, D.E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addision-Wesley, 1989